About Strongly Universal Cellular Automata

Maurice Margenstern,
Université de Lorraine,
LITA EA3097,
Campus du Saulcy,
57045 Cédex, France
e-mail: maurice.margenstern@univ-lorraine.fr,\margenstern@gmail.com

April 24, 2013

This paper is dedicated to Professor Phan Dinh Dieu

Abstract

In this paper, we construct a strongly universal cellular automaton on the line with 14 states and the standard neighbourhood. We embed this construction into several tilings of the hyperbolic plane and of the hyperbolic $3D$ space giving rise to strongly universal cellular automata with 10 states.

1 Introduction

Many paper about universality of cellular automata, especially those which try to minimize the number of states preserving this property, consider what is called **weakly universal** cellular automata. This term means that the initial configuration of the cellular automaton may be infinite, provided the following requirement is satisfied: the initial configuration must be periodic outside a bounded domain. In the case of the one dimensional line, it is accepted that the periodicity in the left-hand side infinite part is different from the periodicity of the right-hand side one.

In this paper, we consider deterministic cellular automata on the line with the **standard neighbourhood** which means, for each cell, the cell itself and its left- and right-hand side neighbours. For this class of automata, we consider the construction of a **strongly** universal cellular automaton. This restriction means that the initial configuration of the cellular automaton must be finite. There are very few papers in this line and the single one we know, apparently an important one, is the paper [2] where it is claimed that such a cellular automaton with 7 states is constructed. Of course, the reader may ask what is the use of a strongly universal cellular automaton with a bigger number of states. The reason is that, as mentioned in [5], this automaton is not strongly universal. At least, [2] does not prove that the cellular automaton constructed in the paper is strongly universal. Indeed, the construction of [2] is based on a close simulation of a universal Turing machine with 7 states and 4 letters constructed by H. Minsky, see [7]. However, the machine used by [2] is not universal, as pointed out by [8, 9]. Moreover, the initial configuration of the automaton of [2] is infinite, although its structure is very simple. It seems that the authors of [2] do not pay much significance to the distinction between strong and weak universality, a common attitude at that time. Now, if we replace this machine by that of [8, 9] with also 7 states and 4 letters, and using the idea of the simulation performed by [2], we could not obtain a cellular automaton with 7 states but only with 9 states and we would remain with a weakly universal cellular automaton. This number of states is much more than the two states obtained in [1], but the result is obtained significantly more easily.

Several reason explain this situation, but this needs to enter into finer details of the construction performed in both [2] and the present paper.

In Section 2, we give the principles on which is based the construction. In Section 3, we give the execution performed by the automaton, proving that it simulates the Turing machine. In Section4, we construct a strongly universal cellular automaton on the line with 9 states. In Section 5, we extend the result to cellular automata in hyperbolic spaces.

2 Implementing a Turing machine in a one-dimensional cellular automaton

Weak universality results for cellular automata are rather easy to obtain from the simulation of a Turing machine. The idea is to embed the Turing tape into the cellular automaton by regularly putting the symbols of the Turing machine, two neighbouring symbols of the tape being separated by the same number of blanks of the cellular automaton.

In order to obtain a strongly universal cellular automaton, only a finite part of the Turing tape can be embedded in such a way. In order to perform the computation, especially if the computation turns out to be infinite, we have also to implement the continuation of the Turing tape. For the present moment, let us ignore this point to which we go back in Section 5

Ignoring this continuation constraint, it is known that the simulation can be performed with roughly nm-states if the Turing machine has n states and m letters, and it is not difficult to improve this result to $m+2n$ states. In [2], the authors improve this result to $m+n+2$ states. Next, they construct a cellular automaton with seven states and the usual neighbourhood which they call universal. From now on, we denote this automaton by $\mathcal{LN}$.

Let us first indicate how $\mathcal{LN}$ works. We say that a cell is **blank** if it is in the quiescent state. Accordingly, if a cell is blank as well as its left- and right-hand side neighbours, it remains blank at the next step of the computation.

The Turing tape is implemented by inserting the squares of the tape at regular places of the cellular automaton, leaving the same amount of blank cells in between two consecutive places. It is required in [2] that this number should at least be 2. At this point, five states are used: the four letters of the Turing machine, 0, 1, y and A, where 0 is the blank of the Turing machine, and the blank state which we denote by _. Then, the instructions of the Turing machine are analyzed according to the states and the motion of the head induced by the state. In principle, a state might possess instructions which trigger a move to left as well as instructions which provide a move to right. However, it can be noticed in the table of Minsky's Turing machine, that some states contain instructions always to left while some others contain instructions always to right. This leads to the notion of **impulsion** which is a couple consisting of the state and a direction. *A priori*, a state s triggers two impulsions which we denote by Ls and Rs in self-explaining notations. Table 1 displays the program of Minsky's Turing machine. In this table, for each instruction, the move is always mentioned, halting instruction excepted, for the new letter and the new state, this new element is mentioned only if it differs from the corresponding current one.

TABLE **1** *Table of Minsky's Turing machine with* 7 *states and* 4 *letters.*

	0	1	y	A
1	L	L2	0L	1L
2	yR	AR	0L1	yR6
3		AL	L	1L4
4	yR5	L7	L	1L
5	yL3	AR	R	1R
6	AL3	AR	R	1R
7	yR6	R	0R	0R2

From the table, we can see that there are 4 right-hand side impulsions: R2, R5, R6 and R7; and 5 left-hand side ones: L1, L2, L3, L4 and L7. We encode a left-hand side impulsion by two consecutive cells in the form T x and a right-hand side one in the form x T. As we have four letters in the Turing machine, we can use them for encoding the right-hand side impulsions taking x in {0,1,y,A} and an additional letter B for taking x in {0,1,y,A,B} for the left-hand side impulsions the substitutions being performed as in the above order.

The simulation of the Turing machine works as follows.

Call **background** a configuration of the cellular automaton of the form (_ x _)$^\infty$ where x in {0,1,b,c} and x = 0 for all x outside a finite interval of the support of the cellular automaton. Call these letters the **Turing symbols**. A **key** configuration is a configuration where there is at most one T in the situation (_ x _)$^\infty$ y T (_ x _)$^\infty$ or (_ x _)$^\infty$ T y (_ x _)$^\infty$. It is assumed that there is at most one single cell in T in each key configuration. In fact, there may be more blanks in between consecutive Turing symbols in a key configuration. Rules are assumed which allow y T to move to the right and x T to move to the left until T stands at distance 1 from a Turing symbol: we have T _ x or x _ T. We say that in both cases we have a **collision**. In order to prove the simulation of Minsky's Turing machine by $\mathcal{L}N$, it is enough to prove that the transitions of Table 2 can be managed by the rules when there is a collision.

TABLE **2** *The collisions associated to Minsky's Turing machine with* 7 *states and* 4 *letters in* [2].

0 T _ 0 T 1 _ y	0 T _ 1 A _ 0 T	0 T _ y y _ 0 T	0 T _ A 1 _ 0 T
1 T _ 0 T 1 _ A	1 T _ 1 A _ 1 T	1 T _ y y _ 1 T	1 T _ A 1 _ 1 T
y T _ 0 y _ y T	y T _ 1 A _ y T	y T _ y T y _ 0	y T _ A y _ 1 T
A T _ 0 y _ 1 T	A T _ 1 1 _ A T	A T _ y 0 _ A T	A T _ A 0 _ y T
0 _ T 0 y _ y T	1 _ T 0 A _ y T	y _ T 0 T y _ 0	A _ T 0 y _ 1 T
0 _ T 1 _ 0 _	1 _ T 1 T 1 _ A	y _ T 1 T 1 _ y	A _ T 1 T A _ 1
0 _ T y T y _ 0	1 _ T y T 0 _ 1	y _ T y T y _ 0	A _ T y T y _ 1
0 _ T A y _ 0 T	1 _ T A T B _ 1	y _ T A T A _ y	A _ T A T A _ 1
0 _ T B y _ 1 T	1 _ T B 1 _ A T	y _ T B 0 _ A T	A _ T B 0 _ y T

We leave the reader with this checking, referring him/her to [2].

3 The simulation of the present paper

Our plan is to follow the same path by changing Minsky's machine to Rogozhin's machine with 7 states and 4 letters. The latter machine has been proved to be strongly universal in [8, 9]. Table 3 displays the program of Rogozhin's machine in a similar way as Table 1 does for Minsky's machine. This time we have 4 left-hand side impulsions: L1, L4, L6, L7 respectively encoded T 0, T 1, T b and T c. There are now 5 right-hand side instructions: R1, R2, R3, R5 and R6 respectively encoded 0 T, 1 T, b T, c T and D T.

Table 3 *Table of Rogozhin's Turing machine with 7 states and 4 letters.*

	0	1	b	c
1	L	$0L$	$cR2$	bL
2	$1R$	$0L1$	cR	$1R5$
3	$1L4$	R	cR	$1R5$
4	$1L7$	L	cL	bL
5	$cL4$	R	cR	bR
6	$R5$	$0R$	R	$0R1$
7	$R3$	1	$L6$	c

Table 4 *The expected collisions induced by Rogozhin's Turing machine with 7 letters and 4 states.*

$\frac{\texttt{0 \_ T 0}}{\texttt{T 0 \_ 0}}$	$\frac{\texttt{1 \_ T 0}}{\texttt{T 0 \_ 0}}$	$\frac{\texttt{b \_ T 0}}{\texttt{c \_ 1 T}}$	$\frac{\texttt{c \_ T 0}}{\texttt{0 \_ b T}}$
$\frac{\texttt{0 \_ T 1}}{\texttt{T c \_ 1}}$	$\frac{\texttt{1 \_ T 1}}{\texttt{T 1 \_ 1}}$	$\frac{\texttt{b \_ T 1}}{\texttt{1 \_ c T}}$	$\frac{\texttt{c \_ T 1}}{\texttt{1 \_ b T}}$
$\frac{\texttt{0 \_ T b}}{\texttt{0 \_ c T}}$	$\frac{\texttt{1 \_ T b}}{\texttt{0 \_ D T}}$	$\frac{\texttt{b \_ T b}}{\texttt{b \_ D T}}$	$\frac{\texttt{c \_ T b}}{\texttt{0 \_ 0 T}}$
$\frac{\texttt{0 \_ T c}}{\texttt{0 \_ b T}}$	$\frac{\texttt{1 \_ T c}}{\texttt{\_ b \_}}$	$\frac{\texttt{b \_ T c}}{\texttt{T b \_ D}}$	$\frac{\texttt{c \_ T b}}{\texttt{\_ c \_}}$
$\frac{\texttt{0 T \_ 0}}{\texttt{T 0 \_ 0}}$	$\frac{\texttt{0 T \_ 1}}{\texttt{T 0 \_ 0}}$	$\frac{\texttt{0 T \_ b}}{\texttt{c \_ 1 T}}$	$\frac{\texttt{0 T \_ c}}{\texttt{0 \_ b T}}$
$\frac{\texttt{1 T \_ 0}}{\texttt{1 \_ 1 T}}$	$\frac{\texttt{1 T \_ 1}}{\texttt{T 0 \_ 0}}$	$\frac{\texttt{1 T \_ b}}{\texttt{c \_ 1 T}}$	$\frac{\texttt{1 T \_ c}}{\texttt{1 \_ c T}}$
$\frac{\texttt{b T \_ 0}}{\texttt{T 1 \_ 1}}$	$\frac{\texttt{b T \_ 1}}{\texttt{1 \_ b T}}$	$\frac{\texttt{b T \_ b}}{\texttt{c \_ b T}}$	$\frac{\texttt{b T \_ c}}{\texttt{b \_ b T}}$
$\frac{\texttt{c T \_ 0}}{\texttt{T 1 \_ c}}$	$\frac{\texttt{c T \_ 1}}{\texttt{1 \_ c T}}$	$\frac{\texttt{c T \_ b}}{\texttt{c \_ c T}}$	$\frac{\texttt{c T \_ c}}{\texttt{b \_ c T}}$
$\frac{\texttt{D T \_ 0}}{\texttt{0 \_ c T}}$	$\frac{\texttt{D T \_ 1}}{\texttt{0 \_ D T}}$	$\frac{\texttt{D T \_ b}}{\texttt{b \_ D T}}$	$\frac{\texttt{D T \_ c}}{\texttt{0 \_ 0 T}}$

Accordingly, although the global display of the data on the tape is the same for both Minsky's and Rogozhin's Turing machines, there working are very different as shown by the opposite structure of the impulsions: there are more of them going to left in Minsky's machine while more of them are going to right in Rogozhin's machine.

Table 4 shows us the collision conditions which must be satisfied by rules implementing Rogozhin's machine, following the same implementation idea as the one indicated in [2]. This time we are sure to get a true universal Turing machine, as Rogozhin's machine is free from the defect of Minsky's machine which erases its tape just before halting when the computation halts from the starting configuration. However, the cellular automaton we obtain is still weakly universal.

Before turning to the way to bypass this problem, let us see how rules can be devised to satisfy the conditions displayed by Table 4.

In order to do this, we display the successive configurations around a collision until the resulting configuration is obtained in Tables 5 and 6.

At first glance, these executions are alike those of [2]. Before looking at the difference of these tables with those of [2], let us have a closer look at the working of the simulation.

Table **5** *Execution of the rules for the collisions of a left-hand side impulsion with a Turing letter.*

```
impulsions to left:

  . . 0 . T 0 . .    . . 1 . T 0 . .    . . b . T 0 . .    . . c . T 0 . .
  . . 0 D 0 . . .    . . 1 0 0 . . .    . . b D 0 . . .    . . c c 0 . . .
  . . G 1 b . . .    . . G 1 0 . . .    . . 0 b b . . .    . . c D 1 . . .
  . . T 0 1 . . .    . . T 0 1 . . .    . . 1 1 T . . .    . . b G 1 . . .
  . T 0 . 0 . . .    . T 0 . 0 . . .    . . c . 1 T . .    . . T 0 c . . .
                                                           . T 0 . b . . .

  . . 0 . T 1 . .    . . 1 . T 1 . .    . . b . T 1 . .    . . c . T 1 . .
  . . 0 D 1 . . .    . . 1 0 1 . . .    . . b D 1 . . .    . . c c 1 . . .
  . . G c 1 . . .    . . G c 0 . . .    . . 0 c 1 . . .    . . c D c . . .
  . . 0 c c . . .    . . 0 b 1 . . .    . . T 1 c . . .    . . b G b . . .
  . . T c 0 . . .    . . 1 c D . . .    . T 1 . c . . .    . . T 1 D . . .
  . T c . 1 . . .    . . T 1 b . . .                       . T 1 . b . . .
                     . T 1 . 1 . . .

  . . 0 . T b . .    . . 1 . T b . .    . . b . T b . .    . . c . T b . .
  . . 0 D b . . .    . . 1 0 b . . .    . . b D b . . .    . . c c b . . .
  . . G c T . . .    . . G D T . . .    . . 0 D T . . .    . . c 0 T . . .
  . . 0 . c T . .    . . 0 . D T . .    . . G . D T . .    . . 0 . 0 T . .
                                        . . b . . D T .

  . . 0 . T c . .    . . 1 . T c . .    . . b . T c . .    . . c . T c . .
  . . . 0 D c . .    . . 1 0 c . . .    . . b D c . . .    . . c c c . . .
  . . . G c b . .    . . G T b . . .    . . 0 1 b . . .    . . c T 0 . . .
  . . . 0 b T . .    . . . b . . . .    . . 1 c 1 . . .    . . . c . . . .
  . . . 1 G b T .    . . . b . . . .    . . T b c . . .    . . . c . . . .
  . . . 0 . . b T    . . . b . . . .    . T b . b . . .    . . . c . . . .
```

An attentive look at the collisions show us that when the direction of the resulting impulsion of the collision is the same as the direction of the incident impulsion, the place of the new Turing symbol is displaced with respect of that of the current one by two cells in the opposite directions. When the directions are opposite, this mean that we have a half-turn of the Turing head, the new symbol replaces the previous one at the same place.

This means that after a half-turn is performed, the Turing symbols collided by the impulsion take the place of the symbols before it was placed by the impulsion arriving at a half-turn.

Table **6** *Execution of the rules for the collisions of a right-hand side impulsion with a Turing letter.*

```
  impulsions to right:

    . 0 T . 0 . . .   . 0 T . 1 . . .   . 0 T . b . . .   . 0 T . c . . .
    . . 0 b 0 . . .   . . 0 1 1 . . .   . . 0 b b . . .   . . 0 1 c . . .
    . . 1 1 b . . .   . . 1 1 b . . .   . . 1 1 T . . .   . . 1 G c . . .
    . . c 0 1 . . .   . . c 0 1 . . .   . . c . 1 T . .   . . 0 c G . . .
    . . 0 0 0 . . .   . . 0 0 0 . . .                     . . T 0 G . . .
    . . G 1 0 . . .   . . G 1 0 . . .                     . T 0 . b . . .
    . . T 0 1 . . .   . . T 0 1 . . .
    . T 0 . 0 . . .   . T 0 . 0 . . .

    . 1 T . 0 . . .   . 1 T . 1 . . .   . 1 T . b . . .   . 1 T . c . . .
    . . 1 b 0 . . .   . . 1 1 1 . . .   . . 1 b b . . .   . . 1 1 c . . .
    . . 1 H b . . .   . . c H b . . .   . . 1 1 T . . .   . . c H c . . .
    . . D H 0 . . .   . . 0 0 0 . . .   . . c . 1 T . .   . . 0 D G . . .
    . . 0 1 T . . .   . . G 1 0 . . .                     . . G G T . . .
    . . 1 . 1 T . .   . . T 0 1 . . .                     . . 1 . c T . .
                      . T 0 . 0 . . .

    . b T . 0 . . .   . b T . 1 . . .   . b T . b . . .   . b T . c . . .
    . . b b 0 . . .   . . b 1 1 . . .   . . b b b . . .   . . b 1 c . . .
    . . c G b . . .   . . H c b . . .   . . c b T . . .   . . H 1 c . . .
    . . 0 c D . . .   . . 1 b T . . .   . . c . b T . .   . . 1 H c . . .
    . . T 1 b . . .   . . 1 . b T . .                     . . D D G . . .
    . T 1 . 1 . . .                                       . . H b T . . .
                                                          . . b . b T . .

    . c T . 0 . . .   . c T . 1 . . .   . c T . b . . .   . c T . c . . .
    . . c b 0 . . .   . . c 1 1 . . .   . . c b b . . .   . . c 1 c . . .
    . . c 1 b . . .   . . 0 G b . . .   . . c c T . . .   . . 0 G c . . .
    . . 0 c 1 . . .   . . 0 D D . . .   . . c . c T . .   . . 0 H G . . .
    . . T 1 c . . .   . . G G T . . .                     . . D G T . . .
    . T 1 . c . . .   . . 1 . c T . .                     . . b . c T . .

    . D T . 0 . . .   . D T . 1 . . .   . D T . b . . .   . D T . c . . .
    . . D b 0 . . .   . . D 1 1 . . .   . . D b b . . .   . . D 1 c . . .
    . . 0 D b . . .   . . 1 0 b . . .   . . 0 D T . . .   . . 1 D c . . .
    . . G c T . . .   . . G D T . . .   . . G . D T . .   . . c c b . . .
    . . 0 . c T . .   . . 0 . D T . .   . . b . . D T .   . . c 0 T . . .
                                                          . . 0 . 0 T . .
```

Tables 7 and 8 allow us to understand what happens. Say that a configuration by the end of a collision is a **standard** one if it is of the form `z .  y T` or `T y .  z`, where `z` is the new state after the collision which corresponds to the execution of the current instruction of the Turing machine on the currently scanned symbol. From these tables, we can see that if we consider three consecutive Turing symbols where the middle one is scanned by an impulsion then, after the collision is completed, the positions of the extremal Turing symbols are unchanged. This ensures that the working of the cellular automaton faithfully simulates the Turing machine. Now, there are a few exceptional situations illustrated by Table 9, where the new state of the scanned symbol appears one step later with respect to a standard configuration. This can be seen in Table 5 for two impulsions whose initial configuration around a collision are `b .  T b` and `0 .  T c`. Note that in both cases, we have to deal with a half-turn of the Turing machine. In Table 6, there is a single exception: the collision `D T . b`. Note that in this case, the result is the same impulsion and again, the new state appears one step later with respect to the standard configuration.

TABLE **7** *Standard situation for an impulsion whose direction is not changed by the collision. Left-hand side: impulsion to right; right-hand side: impulsion to left.*

. u . . . . v . . x . . y . .	. u . . v . . . . x . . y . .
. u . a T . v . . x . . y . .	. u . . v . T a . x . . y . .
. u . . w . b T . x . . y . .	. u . T b . w . . x . . y . .

TABLE **8** *Situations of a standard half-turn. Left-hand side: half-turn to left; right-hand side: half-turn to right.*

. u . . . . v . . x . . y . .	. u . . v . . . . x . . y . .
. u . a T . v . . x . . y . .	. u . . v . T a . x . . y . .
. u . T b . w . . x . . y . .	. u . . w . b T . x . . y . .

TABLE **9** *In these cases, there is delay in the transformation of the new state by one step. Left-hand side: for an impulsion to right not changed by the collision; right-hand side: for a half-turn to right. There are no other cases involving such a delay.*

. u . . . . v . . x . . y . .	. u . . . . v . . x . . y . .
. u . a T . v . . x . . y . .	. u . . v . T a . x . . y . .
. u . . w . b T . x . . y . . . u . . z . . b c x . . y .	. u . . z . b T . x . . y . . . u . . w . . b c x . . y . .

Now, from Table 9, we can see that again, the positions of the two extremal Turing symbol in a configuration of three consecutive such symbols are unchanged after the collision is performed. Note that in the exceptional situation, when we look at the final time of the collision, the next one already started: it is at its first step.

Now, the initial configuration can be devised in such a way that at the starting top of the computation, the configuration delimited by the two neighbouring Turing symbols of the initially scanned cell is conform to an initial configuration of Tables 7, 8 or 9.

Let us call $\mathcal{M}$ the cellular automaton which we constructed in this section.

4 Strongly universal cellular automata on a one-dimensional line

As mentioned in Section 3, the automaton $\mathcal{M}$ is not yet strongly universal: as $\mathcal{LN}$, it works on a background which is not finite.

We can improve $\mathcal{M}$ by providing a construction of the background during the computation of the automaton. There is a very simple way to do that by using three additional states, say `#`, `$` and `&`.

First, as the considered Turing machine simulates the computation of a tag-system, we refer the reader to [7, 9] for this notion, we may assume that the computation is performed on a tape which is infinite on one direction only. This means that there is a square c of the Turing tape such that the head of the machine never goes to the left of c. Consequently, to the left of the cell corresponding to the position of this c on the tape, we can consider that all the cells are blank. Initially, the cells of the automaton must also be blank to the right of a cell d. Now, we may chose the position of d. We shall fix it a bit later, but we already can consider that d is placed at a such position that the interval $[c, d]$ contains the image of the initial configuration of the Turing tape under the implementation described in Section 2. Now, let us consider the last Turing symbol `0` which is within $[c, d]$. It may be assumed that the head does not look at it. We can consider that to the right of this `0`, all cells are blank.

Second, replace `0` by `#`. Table 10 provides us with a scheme of execution which shows how it is possible to construct the background needed for $\mathcal{M}$ during its computation :

Table **10** *Using three states to construct the background for* $\mathcal{M}$

```
. # . . . . . . .
. 0 $ . . . . . .
. 0 . & . . . . .
. 0 . . # . . . .
```

However, it is possible to obtain such a construction without appending new states. This is explained by the computation given in Table 11. The idea is to define a sequence of patterns using the states of $\mathcal{M}$ only which can be deduced from one another and which provide us with the required construction. This is made possible by the fact that the set of rules of $\mathcal{M}$ does not involve all possible triples of states. Moreover, as shown by Table 11, the computation also involves quadruples and even strings of five letters which increase the possibility to obtain the desired effect. The involved rules are given in the appendix.

Table **11** *Using an appropriate pattern to construct the background needed by* $\mathcal{M}$*.*

```
. . . G b D . . . .
. . . 1 G H . . . .
. . . 0 D H D . . .
. . . G G G H . . .
. . . 1 G G H D . .
. . . 0 . G H H . .
. . . 0 . . G b D .
```

Although this sequence of pattern is not very complex, the reader might think that simpler solutions may exist. In fact, this solution was dictated by another constraint which we did not yet explicit: we require the advance of the pattern of Table 11 to be at speed $\frac{1}{2}$.

There are two reasons for this constraint.

The first one is that the construction of the background must be faster than the simulation of the computation of the Turing machine. This is satisfied by the pattern of Table 11. Indeed, Tables 5 and 6 show that the speed of the computation by $\mathcal{M}$ does not exceed $\dfrac{1}{3}$ and sometimes it is only $\dfrac{1}{7}$. Indeed, three steps of computation are needed to reach the next Turing symbol in the fastest situations and seven of them are required in the slowest ones.

The second reason of this constraint is the following. When it happens that the simulated Turing machine halts, the simulating process must also halt. In particular, the construction of the pattern must be stopped. If the progression of the constructing patter would be at speed 1, it would never be reached as the halting usually comes a definite time after the start of the computation. And so, the progression must be at least at speed $\dfrac{1}{3}$, so that $\dfrac{1}{2}$ is a good tempo. Let us again call $\mathcal{M}$ the cellular automaton with 9 states whose rules perform the computation given by Tables 5 and 6, which also allow impulsions to travel freely on an empty background and which perform the construction process depicted by Table 11. The rules of $\mathcal{M}$ are given in Table 13. It is the point to note that Tables 5, 6 and 13 where established with the help of a computer program which checked their coherence. Also, the computer program produced the displays given in these tables.

Now, the second constraint raises another problem which was not considered in [5, 6] as the goal was to work in another space. But this fact was also unseen in [2] as there the cellular automaton works on an infinite configuration. The new problem to which we are now faced is that when the Turing machine halts its computation, its head is at the leftmost square of its configuration. Now, in our simulation, the construction of the background moves to the right. And so, the halting of the computation must trigger a signal which must cross over the configuration without destroying it in order to reach the constructing pattern and stop its action. This latter condition requires to at least replicate the symbols of the tape but, at this stage, when the simulation halts there are only the blank and the Turing symbols on the tape. This makes it easier for the halting signal to recognize the time when it meets the constructing patterns which consists mostly in non-Turing symbols.

Table 12 illustrates the process, assuming that we append five additional states: `S` for the stopping signal itself, `3`, `4`, `u` and `v` for replacing `0`, `1`, `b` and `c` respectively. The table shows how the stopping signal crosses the Turing tape without altering it and how it stops and erases the pattern of the construction of the background. Let us call $\mathcal{Q}$ the cellular automaton obtained from $\mathcal{M}$ by appending the just mentioned new states with the rules allowing the motions indicated by Table 12. Accordingly we proved:

THEOREM **1** *There is a deterministic cellular automaton on the line, with the standard neighbourhood and* 14 *states which is strongly universal.*

Indeed, $\mathcal{Q}$ satisfies the requirements of the theorem.

Table **12** *The work of the stopping signal* S. *Left-hand side: crossing the Turing configuration without altering it. Right-hand side: all the three possible approaches to the stopping of the pattern constructing the background.*

```
. S 0 . . b . . c . . 1 . .
. . 3 . . b . . c . . 1 . .
. . 0 S . b . . c . . 1 . .
. . 0 . S b . . c . . 1 . .
. . 0 . . U . . c . . 1 . .
. . 0 . . b S . c . . 1 . .
. . 0 . . b . S c . . 1 . .
. . 0 . . b . . V . . 1 . .
. . 0 . . b . . c S . 1 . .
. . 0 . . b . . c . S 1 . .
. . 0 . . b . . c . . 4 . .
. . 0 . . b . . c . . 1 S .
. . 0 . . b . . c . . 1 . S
```

```
. . S G b D . . . .
. . . 3 G H . . . .
. . . 0 S H D . . .
. . . 0 . S H . . .
. . . 0 . . S D . .
. . . 0 . . . . . .
. . . 0 . . . . . .
-------------------
. S . G b D . . . .
. . S 1 G H . . . .
. . . 3 D H D . . .
. . . 0 S G H . . .
. . . 0 . S H D . .
. . . 0 . . S H . .
. . . 0 . . . . . .
```

```
3 . . G b D . . . .
0 S . 1 G H . . . .
0 . S 0 D H D . . .
0 . . 3 G G H . . .
0 . . 0 S G H D . .
0 . . 0 . S H H . .
0 . . 0 . . S b D .
0 . . 0 . . . S H .
0 . . 0 . . . . . .
-------------------
0 . . G b D . . . .
3 . . 1 G H . . . .
0 S . 0 D H D . . .
0 . S G G G H . . .
0 . . 4 G G H D . .
0 . . 0 S G H H . .
0 . . 0 . S G b D .
```

Table **13** *Table of the rules for the cellular automaton* $\mathcal{M}$.

.	.	0	1	b	c	D	T	G	H
..	.	.	.	.	.	.	T	.	.
0.	.	.		.	.	.	D	.	
1.	.		.	.	.		0		
b.	.			.	.	.	D		
c.	.		.	.	.		c		
D.	.								
T.	T	b	1	b	1				
G.	.					.			
H.	D								

0	.	0	1	b	c	D	T	G	H
.0	0	G	1	1	T	G	.	0	D
00	0	1							
10	1	1	c	D	T				
b0	b								
c0	1		0				.		
D0	b								
T0	.		.		.			.	
G0									
H0	T								

1	.	0	1	b	c	D	T	G	H
.1	1	G	c	1	T	c	.	0	D
01	0		1	c	G		.		
11	b		H	0	H		.		
b1	D		c		1				
c1	c		G	c	G				
D1	1		0		D				
T1	.			.	.	.			
G1	c	0		0					
H1					H				

b	.	0	1	b	c	D	T	G	H
.b	b		H	c		0	.	T	
0b	T	1	c	1			G		
1b	1	H		1			.		
bb	T	G		b					
cb	T	1		c			.		
Db	T	D		D					
Tb	.				.				
Gb	D					G	.		
Hb	0						.		

c	.	0	1	b	c	D	T	G	H
.c	c	0	0	c	c	b	.	0	0
0c	b		1		c	1		0	
1c	c		b			1			
bc	b								
cc	0	D	D	0	T		.		
Dc	b								
Tc	.	.							
Gc	G	b	c	b			.		
Hc	G			b					

D	.	0	1	b	c	D	T	G	H
.D			1	0		H	.	b	0
0D		1	c	c	c	G	.	G	G
1D	b				c				
bD	H	b	c	D	1				
cD	b		G		G				
DD	T							b	
TD									
GD							.		
HD	H								

T	.	0	1	b	c	D	T	G	H
.T		0	1	b	c				
0T	0								
1T	1								
bT	b								
cT	c	c							
DT	D								
TT									
GT	c			b					
HT									

G	.	0	1	b	c	D	T	G	H
.G	b		T	1	0	0	.	1	.
0G	b			D	H				
1G				.	c			.	D
bG			0	1					
cG	G			c					
DG	T						.		
TG									
GG							.	G	G
HG	T								

H	.	0	1	b	c	D	T	G	H
.H			1	b	1				
0H								G	
1H				H	D				
bH									
cH				0	D				
DH		1				G			
TH									
GH	H					H			G
HH	b								

5 Strongly universal cellular automata in hyperbolic spaces

In this section, we first remind the tilings we consider for the construction of our automata, see Sub-section 5.1. Then, we remind their construction, replicating that of [5] for the convenience of the reader, see Sub-section 5.2. Finally, we checking the implementation of the rules given in Tables 13 in this context in Sub-section 5.3.

5.1 Tessellations in hyperbolic spaces

We refer the reader to [3, 6] for an introduction to hyperbolic geometry suited for the purpose of the paper. We just mention that we take Poincaré's disc model for the hyperbolic plane. Our illustrations will take place in this model. We also take granted that this model can be generalized to higher dimension, adapting the terms used for the plane to terms of the corresponding space. In particular, the disc is now a hyper-ball, its border a hypershpere, respectively a ball, and a sphere for the hyperbolic $3D$-space. The lines become hyper-planes, planes in the case of the hyperbolic $3D$-space.

Among tilings, an important role is payed by **tessellations**. They are tilings based on a regular convex polygon, a regular convex polyhedron in the $3D$-space. The tiling itself is constructed by the following process: starting from a copy of the basic polygon P, new copies are obtained by reflection of P in its sides and, recursively, of the images in their sides. An important property of hyperbolic geometry is the existence of infinitely many tessellations in the hyperbolic plane, while, in the Euclidean plane we have three only of them, up to similarity. There are only four tessellations in the hyperbolic $3D$-space. We shall consider two tessellations of the hyperbolic plane and one of the hyperbolic $3D$-space.

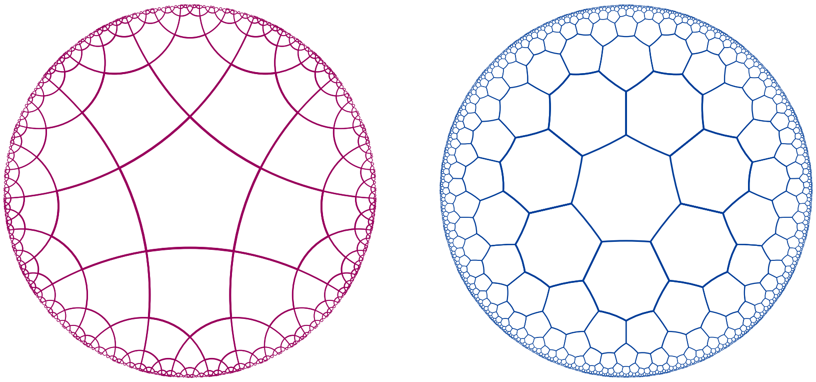


FIGURE 1 *Two tessellations of the hyperbolic plane illustrated in Poincaré's disc. Left-hand side: the pentagrid. Right-hand side, the heptagrid.*

In the hyperbolic plane, we consider the **pentagrid** and the **heptagrid** illustrated by Figure 1. The pentagrid is based on the regular pentagon with right angles while the heptagrid is based on the regular heptagon whose interior angle is $\dfrac{2\pi}{3}$.

In the hyperbolic $3D$-space, we consider the **dodecagrid** based on Poincaré's dodecahedron. This dodecahedron is constructed from the pentagon of the pentagrid in the same way as the cube is constructed from the square. In order to represent this tessellation and taking into account the specificity of our implementation,we shall consider projections on two orthogonal planes. In fact, we need only the tiles which are in contact with one of these planes by one face exactly and we call this plane **horizontal**, denoted by $\mathcal{H}$. We say that the face which is on $\mathcal{H}$ is the **back** of the tile and its **top** is the face which is opposite to the back. Now, the principle of the projection is illustrated in Figure 2 and consists in the following.

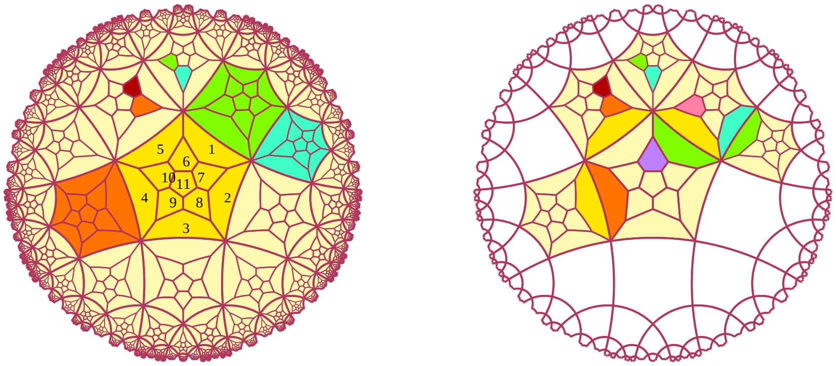


FIGURE **2** *Projection of tiles of the dodecagrid on the horizontal plane. Left-hand side: identifying the tiles. Right-hand side: the convention for colouring the visible faces of a dodecahedron.*

Each tile is projected on its face belonging to the horizontal plane $\mathcal{H}$. For each tile, the projection is performed within the face which lies on $\mathcal{H}$ and it is made from a point, the **projection center** which is on the common perpendicular to both the top and the back of the tile, inside the half-space defined by $\mathcal{H}$ which contains this top. In the right-hand side picture of Figure 2, we decide that a face φ of a tile τ is coloured in γ where γ is the colour associated to the state of the neighbour of τ which shares φ with τ.

As we deal with rotation invariant cellular automata, we do not need to make more precise a system of location of the tiles. However, for checking purposes, where an actual implementation is needed, a convenient system of coordinates for the tiles provides us with a very efficient help. Such a tool was used by the computer program used in [5]. We refer the reader to [3, 4, 6] for a detailed introduction to such a system.

5.2 Implementation

In [5], the author implemented $\mathcal{LN}$ in the pentagrid, in the heptagrid and in the dodecagrid. In that paper, the construction of a line supporting the cellular automaton is performed starting from a finite initial segment of the cells which contains the implementation of the finite initial configuration of the Turing machine. Such a construction can be viewed as an independent cellular automaton $\mathcal{P}$. The simple juxtaposition of both automata, $\mathcal{P}$ and the implementation of $\mathcal{LN}$, lead to a new cellular automaton. The paper mainly succeeded to combine up both automata with a smaller number of states by the identification of some states of $\mathcal{P}$ by those of $\mathcal{LN}$. Now, this assumed that $\mathcal{LN}$ constructs the background it needs which is not the case.

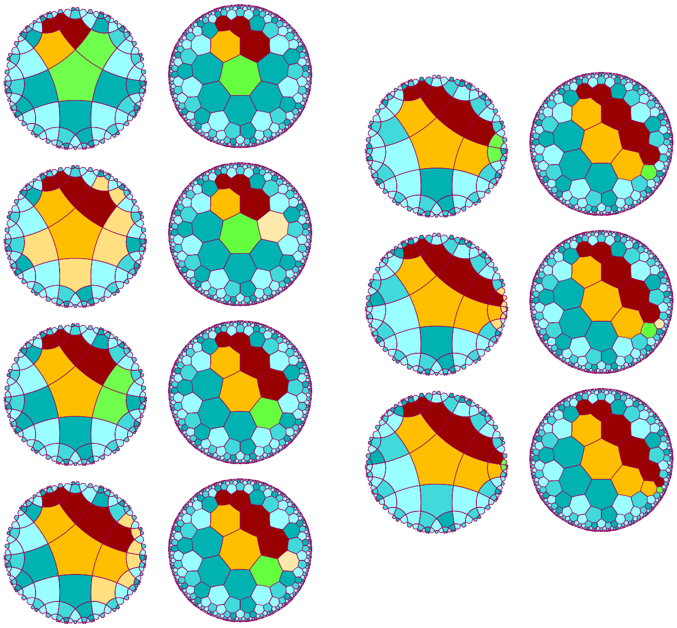


FIGURE **3** *The propagation of the initial segment performed by the automaton $\mathcal{P}$: case of the pentagrid, left-hand side pictures in each column, and of the heptagrid.*

On Figures 3, 4 and 5, we can see the working of the cellular automaton $\mathcal{P}$. The propagation is a very simple construction. It is stopped in two stages. First, the Turing machine halts. Following the description of [9], we can see that this happens when the impulsion `Tc` collides with `c`, see the last row, last column in Table 5. And so, it is enough to replace the rule `c T 0` → `c` by the rule `c T 0` → `dG`, where `dG` is the green state in the mentioned figures: `dG` is the stopping signal in $\mathcal{P}$.

We refer to [6], Chapter 7, Sections 7.2.2 and 7.2.3 for the rules of $\mathcal{P}$ in the pentagrid and in the heptagrid.

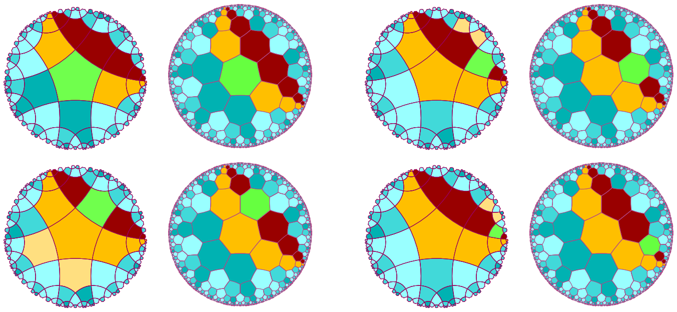


FIGURE **4** *The halting signal arises on the yellow line, goes to the red one and propagates on it as the stopping signal.*

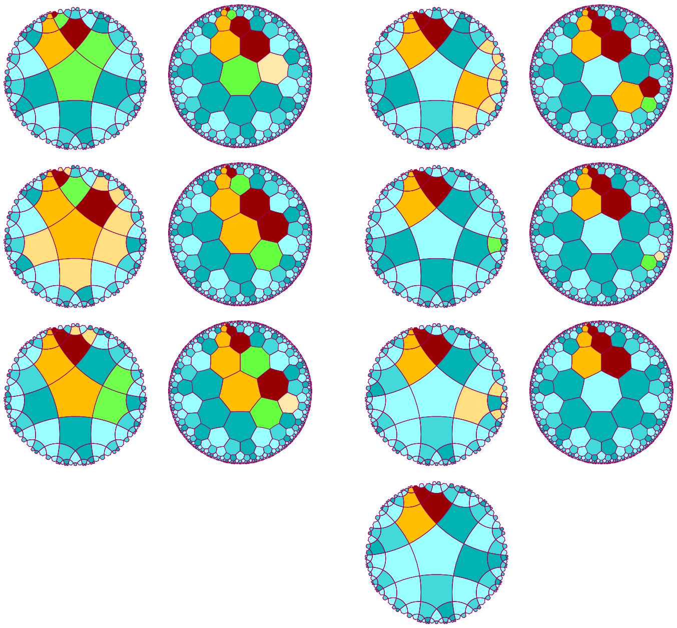


FIGURE **5** *The stopping signal arrives at the end of the segment, erases the constructing pattern and vanishes.*

In the dodecagrid, the construction is again that of [5]. As in that paper, we use an additional state, `M` called mauve colour. As in [5], we four lines are being constructed simultaneously, displayed around a line of the hyperbolic $3D$-space which is a line of the dodecagrid: it completely consists of consecutive edges of tiles belonging to the dodecagrid. As mentioned in Sub-section 5.1, two lines are above $\mathcal{H}$ and the two others are below it. The action of $\mathcal{P}$ in the dodecagrid is illustrated by Figures 6, 7 and 8.

In these figures, we make use of two projections onto $\mathcal{P}$. One is performed from above, the other from below: we may consider that the corresponding projection centers are on the same axis which is orthogonal to $\mathcal{H}$ and to both tops of the tiles having their backs on $\mathcal{H}$. The projection from above is the left-hand side part of the picture which illustrates each step of the computation. The other projection is represented in the right-hand side part of each picture. Also, the right-hand side part of the picture is slightly smaller than the left-hand side part. We also refer to [6], Chapter 7 for the rules of $\mathcal{P}$ in the dodecagrid. Above $\mathcal{H}$, we have the yellow and mauve lines which means that the cells are those of $\mathcal{M}$ in the yellow line and M on the mauve one. Both lines below $\mathcal{H}$ are dark red, denoted by dR.

The actions of the rules are simple: a blank cell which sees a green one, state dG, becomes pink, state dY. A pink cell becomes blank unless it has two pink neighbours through faces sharing an edge: one pink neighbour is above $\mathcal{H}$ and the other is below. In that case, the cell becomes green. At last, the green cell takes the colour of its neighbour on the same line: in particular, if the cell is on the yellow line, it takes the blank of $\mathcal{M}$. Figure 6 illustrates this process.

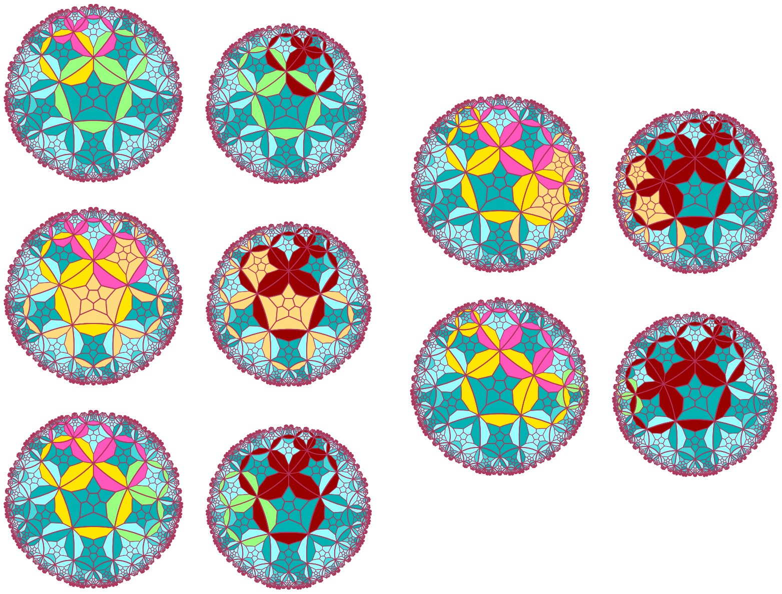


Figure **6** *The propagation of the initial segment performed by the automaton $\mathcal{P}$: case of the dodecagrid.*

Figure 7 indicates what happens when the halting of the simulated Turing machine occurs. As already mentioned, the new rule c T 0 → dG allows the automaton to raise the stopping signal. This rule is transported in the context of the dodecagrid, again see [6], Chapter 7. On Figure 7, it can be seen that the green signal goes to the mauve line and, from there, it moves along the line to the growing end of the segment. This progression is made possible by the fact that the neighbourhood of the two neighbours of the stopping signal on the mauve line are different. The respective neighbourhoods are:

$$\texttt{dR Y M W W dG W}^6, \texttt{dR Y dG W W M W}^6 \tag{a}$$

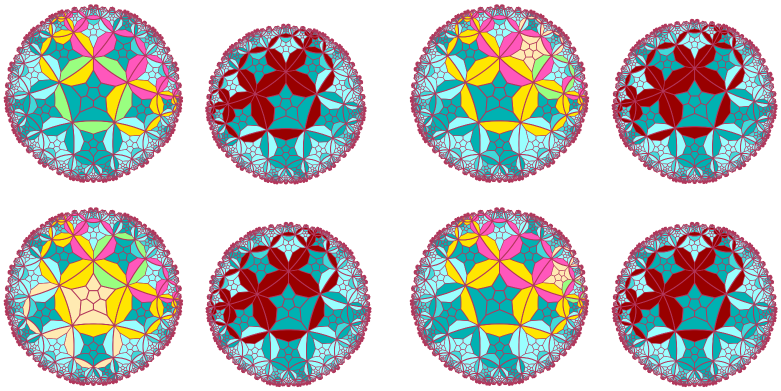


FIGURE **7** *Propagation of the stopping signal when it arose on the yellow line: case of the dodecagrid.*

There could be a confusion if $\texttt{Y} = \texttt{dR}$. Now, we assume presently that $\texttt{Y} \neq \texttt{dR}$. Then, as the red line is visible from face 0 only, the top must be in face 11. Accordingly, the orientation of the neighbours in (a) are different, so that different rules may apply. This justify the fact that this situation allows us to prevent the stopping signal to go to the left-hand side end of the configuration and to force it to go to the right-hand side one, step by step.

Now, going on in this way, the stopping signal arrives to the right-hand side end, which is what Figure 8 illustrates. As shown by the figure, there is always a situation when there is a mauve cell in between the green stopping signal, on its left-hand side, and a pink cell, and on its right-hand side. The mauve cell vanishes, turning to blank and this event step by step disturb the continuation process, allowing us to stop it. Note that, in the case of the plane, the remaining green cell is that of the continuation of the segment. It eventually vanishes to its turn. In the $3D$-space, we remain with three green cells to be destroyed. This is why in the case of the dodecahedron the stopping process takes more time to completely erase the green and the pink cells.

It is important to note that the rules given in [6], chapter 7, for the case of the dodecahedron apply to the cellular automaton defined by $\mathcal{M}$. It is also important to note that $\mathcal{M}$ constructs the periodic background at the same pace as the continuation of the segment: accordingly, the pattern which we defined in Section 3 which is contained in the rules o $\mathcal{M}$ always find blank cells to its right-hand side which allows it to advance at the same step. In this way, the impulsion always find appropriate symbols for its collisions with the Turing symbols. Now, as the stopping signal travels along the mauve line, it may also erase the patterns used to construct the background: the stopping signal is also different from both the mauve and the red states, $\texttt{M}$ and $\texttt{dR}$ respectively.

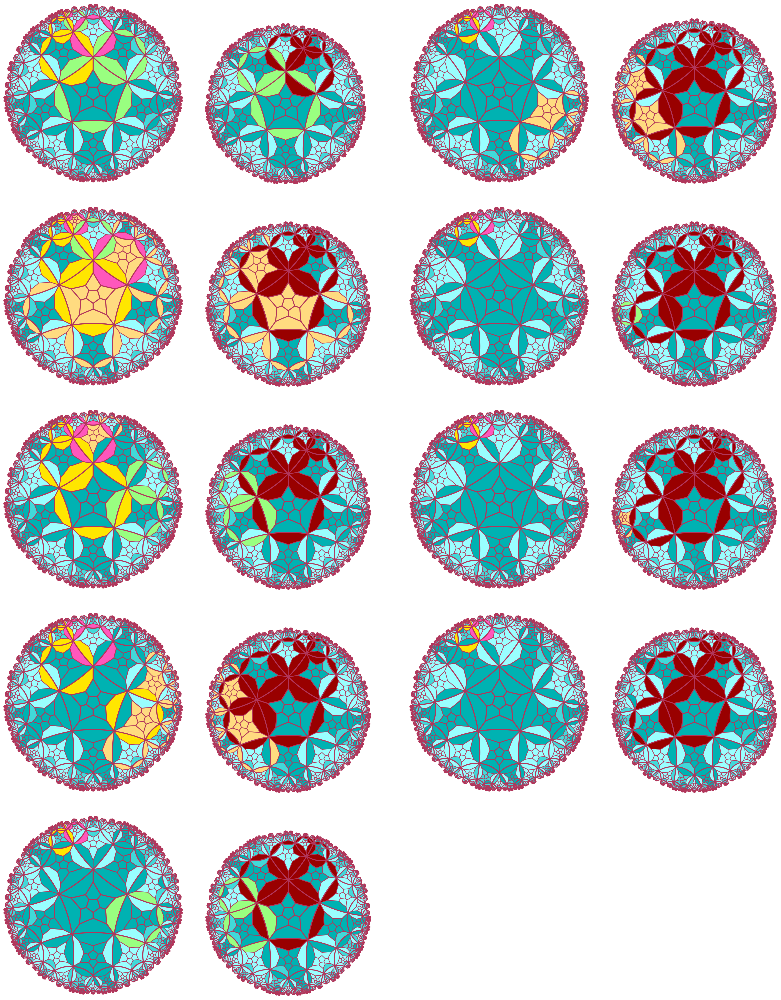


FIGURE **8** *The stopping signal arrives to the other end of the segment, case of the dodecagrid. It erase the constructing pattern and then vanishes.*

5.3 Checking the implementation of $\mathcal{M}$

Now, with $\mathcal{M}$ we have some advantage: it has 9 states, it constructs the background needed for the simulation of Rogozhin's Turing machine and it is actually universal. The only point it does not perform is the halting of the computation. Now, in Sub-section 5.2, we have seen how the cellular automaton $\mathcal{P}$ computes. We have also seen that a simple juxtaposition of both automata solves the problem. In [5], we take advantage of the hyperbolic situation in order to make the combination of $\mathcal{P}$ and $\mathcal{M}$ more compact: the idea is to identify as many states of $\mathcal{P}$ as possible with states of $\mathcal{M}$. Below, (S_0) reminds us the identifications performed in [5] where we used the automaton $\mathcal{LN}$. We can transport this to $\mathcal{M}$ in (S_1) which gives us 11 states immediately.

For the convenience of the reader, we check the identification indicated by (S_1), first in both the pentagrid and the heptagrid and then, in the dodecagrid. In all cases, we consider the structure of the neighbourhood of a cell on each line constructed during the computation. We check that, most often, the neighbourhood allows us to identify to which line the cell belongs and then, which is its correct orientation. When this identification is not possible and leads to several possibilities, we shall see that all of them lead to the application of the same rule, the one which was devised in Sub-section 5.2.

_	0	1	y	A	B	T			
Y	Y	Y	Y	pY	Y	dR	W	dG	(S_0)

where W, Y, dR, G and pY are the states of P.

We may rewrite this as:

_	0	1	b	c	D	T	G	H			
Y	Y	pY	Y	Y	Y	dR	Y	Y	W	dG	(S_1)

where the states of $\mathcal{P}$ are rewritten W, Y, dR, dG and pY in order to distinguish the G of $\mathcal{P}$ from the G of $\mathcal{M}$.

The neighbourhoods for the cells are given in Table 14.

TABLE **14** *Pentagrid and heptagrid: neighbourhoods for cells of the yellow and red lines.*

line		pentagrid	heptagrid
yellow	y	TxWWz	TxWWWzT
red	T	TyTWW	TxyTWWW

Denote by $\mathcal{Q}$ the cellular automaton obtained from $\mathcal{M}$ and $\mathcal{P}$ after applying the identification defined by (S_1). As we require $\mathcal{Q}$ to be deterministic, a cell must know whether it applies a rule of $\mathcal{M}$ or a rule of $\mathcal{P}$ without ambiguity.

The neighbourhoods of Table 14 take into account this identification. Now, we remark that in the pentagrid as well as in the heptagrid, a neighbourhood of a cell of the red line has at least two T's which are separated by at least two blanks. Now, for a cell of the yellow line, the T's are contiguous in the heptagrid and, in the pentagrid there is a single one unless x = T or z = T in which case, the two T's are also contiguous. Note that if a T is replaced by dG or y = dG for a cell of the red line or if x = dG or z = dG for a cell of the yellow line, the neighbourhoods cannot be confused: the same distinction with the place of the W's can be observed. Note that when dG occurs on the yellow line as the stopping signal, its neighbours are never T. More generally, once the stopping signal arose, there is no more T on the yellow line. Also note that when one of the T's in the neighbourhood of a cell of the red line is dG, the two different possibilities give rise to two neighbourhoods which cannot be obtained from one another by rotation around the cell. Now, the neighbourhoods at the end of the growing lines are different. Indeed, we may assume that the process we have seen in Sub-section 5.2 produces cells containing _ on the yellow line: this is required by the pattern which constructs the background needed for the execution of $\mathcal{M}$. As pY is identified with 1, there cannot be any confusion. Also, the propagation rules for dG on the red line match with that of $\mathcal{P}$. And so, the statement of Theorem 2 is proved in the case of the pentagrid and of the heptagrid.

Let us now look at the case of the dodecagrid. From Sub-section 5.2, we know that we have an additional colour: M. We complete (S_1) as follows:

_	0	1	b	c	D	T	G	H			
Y	Y	pY	Y	Y	M	dR	Y	Y	W	dG	(S_2)

where M is identified with D.

This time, Table 15 gives the neighbourhoods of the cells for the four lines we have in this situation.

We can see that, in general, there is no ambiguity. However, in order to check it carefully, we need to have a look on the following problem. In the neighbourhoods of Table 15, there is a pattern denoted by W^6. It consists of face 11 surrounded by a ring of five faces, from face 6 up to face 10, all included, and two additional faces, face 3 and face 4 which are in contact with faces of the ring, namely faces 8, 9 and 10. However, there is another possibility to arrange the same faces in a same decomposition: consider face 9 and the ring of five faces around it: faces 3 and 4, faces 8, 9 and 10, and two faces in contact with the ring: faces 6 and 7, in contact with faces 10, 11 and 8. If the cell considers that its top is 11, we arrive easily to the configuration illustrated by picture (0) in Figure 9: the two faces around face 0 which have blank neighbours fix the position. If the cell considers that its top is face 9, we have then five configurations to examine as there are five rotations leaving the dodecahedron globally invariant and putting face 1 onto $\mathcal{H}$. The five configurations are illustrated by pictures (1) to (5) of Figure 9.

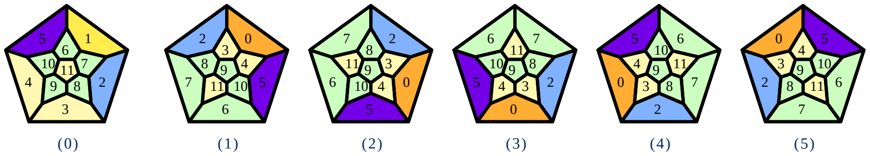
Table **15** *Dodecagrid: neighbourhoods for cells of the yellow, mauve and red lines. Note that* red$_y$*,* red$_m$ *denote the red line below the yellow, mauve line respectively.*

line		
yellow	y	TDzWWxW^6
mauve	D	TyDWWDW^6
red$_y$	T	yTTWWTW^6
red$_m$	T	DTTWWTW^6

Now, we can see on Figure 9 that the single configuration which is compatible with the position of the blank faces 6 and 7 is the configuration illustrated by picture (1). We can notice that then, for a cell of the line, the interpretation of the position of its neighbours on the same line is reversed with respect with the position of the same neighbours when the top is in face 11.

In our proof, we say blank for the blank of $\mathcal{Q}$, denoted by W. For _, we explicitly use the symbol or we speak of the blank of $\mathcal{M}$.

FIGURE **9** *The numbering of the faces of a cell. In* (0)*, the standard numbering, face* 0 *being on the plane* $\mathcal{H}$*. From* (1) *to* (5) *the same numbers for the five rotations leaving the dodecahedron invariant and putting face* 1 *on* $\mathcal{H}$*.*



(0) (1) (2) (3) (4) (5)

On the red lines which are both below $\mathcal{H}$, each cell has three neighbours in T and it is itself in T. As there is at most one cell in T in the yellow line, there are at most two T's among the neighbours of a cell on the yellow or the mauve line. As the cells of the red line never change, this fix their rules easily.

Now, consider a cell of the mauve line. It may hesitate between face 11 or 9 for its top if y $=$ T. However, whatever the choice, it knows that it is on the mauve line as it is in D as well as its faces 2 and 5. Even if one of these two faces is in dG, DDD, dGDD, DdGD and DDdG cannot appear on the yellow line. Accordingly these configurations are those of the mauve line. If we have no neighbour in dG, the cell must remain in D or return to D if it is in dG. If we are after the halting was raised on the yellow line, then there is no more a cell in T in the yellow line. Accordingly, face 1 is recognized as that connected with the yellow line. Otherwise, as face 0 is recognized by the single face in T, if the cell takes face 2 as its face 1, its face 1 should be in contact, on the ring around face 0 to two non-blank faces which is not the case. Accordingly, this choice is impossible as well as the choice of face 5 for a similar reason. And so, the cell recognizes its face 1 which allows to apply the correct rule in the case when dG replaces one of the D's of face 2 and face 5. This allows to apply the correct rules ensuring the motion of dG along the mauve line as the stopping signal.

Consider a cell of the yellow line where there is at most one symbol T. The face 0 is recognized by T. If y $=$ T, x and z are different from T and they are non blank. Face 0 is recognized as the back of the cell and the position of the blanks around it fixes the configuration. If x $=$ T, necessarily y $\neq$ T and z $\neq$ T. If x is chosen as face 0, then face 8 should be the top of the cell but it is not completely surrounded by white faces as face 2 is not blank: it is the face of z. Similarly, if z $=$ T, y $\neq$ T and y $\neq$ T it cannot be taken as face 0: the top would be face 10 which is not possible as it has a non blank face in its neighbouring. And so face 0 is again chosen correctly which forces the cell to recognize face 1 by the position of the blank faces around face 0. If there is no neighbour in T, the single face in T is face 0 and the position of the blank faces around face 0 fixes the correct face 1. Accordingly, a cell of the yellow cell always recognize its face 0 and its face 1 which allows it to apply the appropriate rule of $\mathcal{M}$. It can also be checked on Figure 8, that when the stopping signal arrives, the cancellation process works: the extremal cell of the mauve line vanishes when it can see dG in place of D. Indeed, at this place, the back of the cell is the single visible T and face 1 is the single visible _, the neighbour on the yellow line. Accordingly, dG is located on face 2 while, at the same time face 5 is pink. This erasing of the mauve cell entails, at the next step, that of its yellow neighbour as it has an additional blank neighbour. And then the process goes on as described in [6, 5].

Accordingly, the proof of Theorem 2 is complete.

Theorem **2** *In each of the following tilings: the pentagrid and the heptagrid of the hyperbolic plane, there is a deterministic, rotation invariant cellular automaton with radius* 1 *which has* 11 *states and which is universal.*

Now, we shall prove a stronger theorem: it is possible to get a strongly universal cellular automaton still with one less state.

Note that we cannot identify dG with a state of $\mathcal{M}$: otherwise, dG would go to the red line before the computation is completed, destroying the continuation of the segment and completely perturbing the computation which would have no meaning. Our single possibility is to replace W by a state of $\mathcal{M}$. The most natural way seems to identify the blank state of $\mathcal{M}$ with that of $\mathcal{Q}$.

Now, as the blank of $\mathcal{M}$ is that of $\mathcal{Q}$, this raises a problem at the end of the segment: the transformation of pY into dG requires that the neighbouring yellow cell be non-blank. Accordingly, we have to decide that, at the end of the segment, its continuation provides the same symbol σ for which the rule $\sigma\sigma\sigma \rightarrow \sigma$ should apply. Looking at Table 13, we can see that outside the blank, five states can accept this rule: for b and G, it is already the case and for D, H and T, this can be decided. As b is already identified with M and T is identified with dR, we remain with D, G and H. Now, the choice is also dictated by the pattern which constructs the background for $\mathcal{M}$. As D and H are ending the pattern in several of its steps, we remain with G. It turns out that cancelling the rules concerning the last letter of the pattern and replacing them with the new rules obtained by substituting G to _ in the cancelled rules, we still get a coherent table producing the same collisions and producing a progression of the pattern which is exactly as is expected. Table 16 gives the whole set of rules for $\mathcal{M}$ in this new setting. The new entries bDG, HGG, DGG, HDG, HHG and GHG replace bD., H.., D.., HD., HH. and GH. respectively, taking the same respective outputs.

Note that as the blank is replaced by G on the yellow line in the production of the continuation of the line, the arguments we have seen in the proof of Theorem 2 are still valid.

We only have to check that there is no contradiction between the rules of $\mathcal{M}$and those of $\mathcal{P}$ with the new identification when W replaces a former blank of $\mathcal{M}$ which happens on the yellow line.

First, let us prove that it is possible in the pentagrid and in the heptagrid.

The occurrence of a blank on the yellow line also concerns the red line. Note that on a yellow line, according to what we said in Section 4, namely looking at Tables 7, 8 and 9, we can see that there are never more than two consecutive blanks on the yellow line. It is also easy to observe this constraint at the fixed end of the line. And so, looking at the neighbourhoods of Table 14 concerning the pentagrid and the heptagrid, we have that at most x = W or z = W but not both and, most often, none of them. The distinction between the red line and the yellow one by the number of T's and their separation or not with blanks remains so that in this case, the cells always know to which line they belong. However, they have to know which is their correct orientation. For a cell of the red line, we have always two T's separated by two blanks which are outside the lines, even when one of the T's at most is replaced by dG. The problem concerns the yellow line only.

TABLE **16** *Table of the rules for the cellular automaton $\mathcal{M}$ adapted for $\mathcal{Q}_{10}$.*

.	.	0	1	b	c	D	T	G	H
..	.	.	.	.	.	.	T	.	.
0.	.	.		.	.	.	D	.	
1.	.		.	.	.		0		
b.	.			.	.	.	D		
c.	.		.	.	.		c		
D.	.								
T.	T	b	1	b	1				
G.	.					.			
H.									

0	.	0	1	b	c	D	T	G	H
.0	0	G	1	1	T	G	.	0	D
00	0	1							
10	1	1	c	D	T				
b0	b								
c0	1		0				.		
D0	b								
T0	.		.		.			.	
G0									
H0	T								

1	.	0	1	b	c	D	T	G	H
.1	1	G	c	1	T	c	.	0	D
01	0		1	c	G		.		
11	b		H	0	H		.		
b1	D		c		1				
c1	c		G	c	G				
D1	1		0		D				
T1	.			.	.	.			
G1	c	0		0					
H1					H				

b	.	0	1	b	c	D	T	G	H
.b	b		H	c		0	.	T	
0b	T	1	c	1			G		
1b	1	H		1			.		
bb	T	G		b					
cb	T	1		c			.		
Db	T	D		D					
Tb	.				.				
Gb	D					G	.		
Hb	0						.		

c	.	0	1	b	c	D	T	G	H
.c	c	0	0	c	c	b	.	0	0
0c	b		1		c	1		0	
1c	c		b			1			
bc	b								
cc	0	D	D	0	T		.		
Dc	b								
Tc	.	.							
Gc	G	b	c	b			.		
Hc	G			b					

D	.	0	1	b	c	D	T	G	H
.D			1	0		H	.	b	0
0D		1	c	c	c	G	.	G	G
1D	b				c				
bD	H	b	c	D	1			H	
cD	b		G		G				
DD	T							b	
TD									
GD							.		
HD								H	

T	.	0	1	b	c	D	T	G	H
.T		0	1	b	c				
0T	0								
1T	1								
bT	b								
cT	c	c							
DT	D								
TT									
GT	c			b					
HT									

G	.	0	1	b	c	D	T	G	H
.G	b		T	1	0	0	.	1	.
0G	b			D	H				
1G				.	c			.	D
bG			0	1					
cG	G			c					
DG	T						.	G	
TG									
GG	G						.	G	G
HG	T							D	

H	.	0	1	b	c	D	T	G	H
.H			1	b	1				
0H								G	
1H				H	D				
bH									
cH				0	D				
DH		1				G			
TH									
GH						H		H	G
HH								b	

And so we have the neighbourhoods TxWWz, TxWWWzT in the pentagrid, heptagrid respectively. We have a possible indetermination if x = T or z = T. Assume that x = T. Then if z $\neq$ W, the order while counter-clockwise turning around the cell shows that the T which is the closest to z must be on the red line: otherwise, there are no two, three consecutive W's outside the lines depending on where we are, in the pentagrid, the heptagrid respectively. And so, we get the correct interpretation. Now, if z = W, both possible choices for T with respect to the red line lead to both configurations TWW or WWT on the yellow line at the considered cell. But in both cases, the new state of the cell must be T so this indetermination does not lead to a confusion: the right rule will be applied. We have a similar argument if $\mathtt{z} = \mathtt{T}$ due to the symmetry of the situation. And so, if both $\mathtt{x} \neq \mathtt{T}$ and $\mathtt{z} \neq \mathtt{T}$, then the single T, the two consecutive T's in the pentagrid, the heptagrid respectively, necessarily belong to the red line and the cell knows which rule must be applied. Accordingly, Theorem 3 holds in the case of the pentagrid and that of the heptagrid.

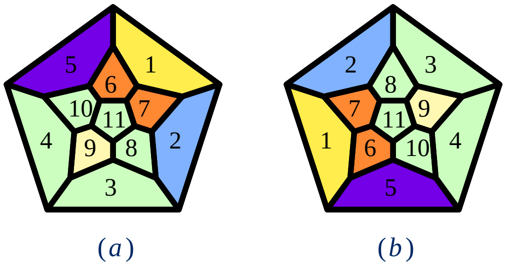
Let us now look at the situation for the dodecagrid.

We have to introduce a new element in our construction. Indeed, consider a cell of the yellow line. From Table 15, the usual neighbouring is $\mathtt{TDzWWxW}^6$, and we have to look at what happens if $\mathtt{x} = \mathtt{D}$ and $\mathtt{z} = \mathtt{W}$. In this case, the back is identified by the unique face in T but the cell may hesitate between two faces for face 1 as we have two faces around the back in D. If we take one choice, we get the configuration DyW on the yellow line and with the other choice, we get the configuration WyD. Now, in general, the corresponding rules of $\mathcal{M}$ are different.

A possible solution consists in appending non-blank neighbours to the neighbourhood of a cell of a line so that both the back and the wall of the cell are uniquely determined.

Indeed, let us go back to the process of constructing the lines: when the green cell occurs at the end of the line by transformation of a pink cell, its blank neighbours become pink. If such a pink cell has itself two pinks neighbours, it becomes green at the next top of the clock. Now, a pink cell may have no pink neighbour as well as a single one. Consider two cells of two lines sharing a face, as illustrated in Figure 10. On the figure, cells (a) and (b) are separated in order to better see the situation of each one. Now, faces 1 are the same face. Accordingly, the neighbour of (a) through its face 6, 7 shares a face with the neighbour of (b) through faces 7, 6 respectively. This means that if (a) and (b) are both green, the pink neighbours corresponding to these faces have, pairwise a common face. Note that for the other faces of (a) and (b) which do not see a cell of a line, their pink neighbours have no pink neighbour.

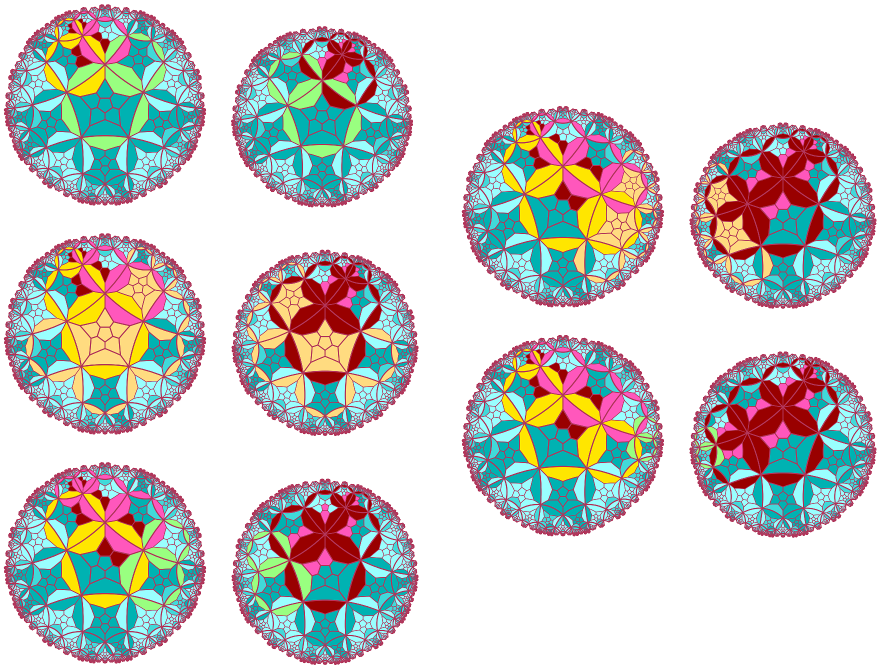
FIGURE **10** *Two faces which share the same face* 1*:* (*a*) *on a line,* (*b*) *on another one. Face* 6, 7 *of* (*a*) *are opposite to faces* 7, 6 *respectively of* (*b*)*. Note that if* (*a*) *is on the yellow line, either its face* 2 *or* 5 *may be blank, but not both at the same time.*



From this, we decide that if a pink cell π has exactly one pink neighbour, it does not become blank. Note that, by construction, the back of π is on a cell of a line. And so, if the back of π is mauve or yellow, π becomes T. If the back of π is red, π becomes D. Say that these faces are **signaling**. We also decide that these new cells remain unchanged. As they have at least nine blank neighbours, they cannot be confused with other cells of the same colour. Now, note that the green cells which are at the end of the line have no signaling faces. However, as they have nine blank neighbours, they cannot be confused with other cells of the lines. We know that they take the colour of their single non-green and non-blank neighbour. After that, the green cell becomes a cell of the line but it is covered with pink faces which also make a distinction with other faces. At the next time, the signaling faces are present so that the general pattern we indicate is in action. Note that the stopping signal is always present in a cell which has the signaling faces, so that it cannot be confused with the green cells which are at the end of the configuration.

Figure 11 illustrates the propagation scheme under the new conditions. Figure 12 illustrates the propagation of the stopping signal in the new setting. Figure 13 shows how the continuation is completed by the stopping signal.

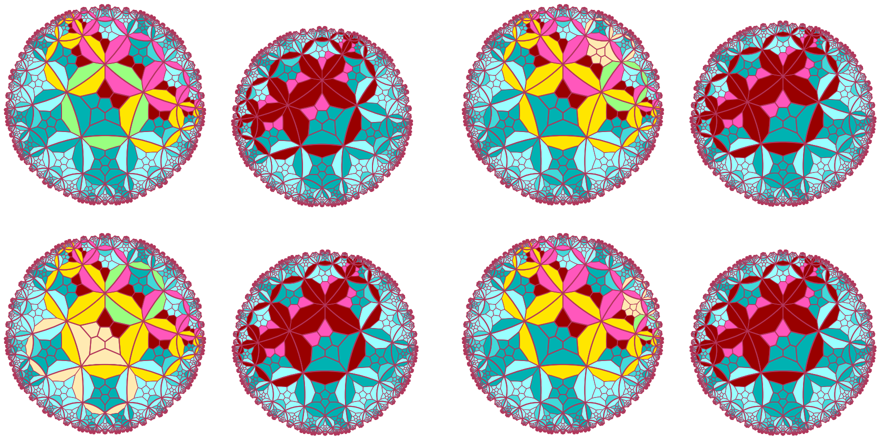
Figure **11** *Propagation of the lines. Note how the signaling faces appear on each line.*



With these new rules, for each cell of a line, exactly one blank face φ is surrounded by blank faces as there can be no more than one blank face among those which surround the back, again see Figure 10. This determines the wall as the face which is opposite to φ in the dodecahedron. On the figure, face φ

is face 9. Now, faces 6 and 7 are the neighbours of faces 1 and 11, the wall and the top, by two edges.

FIGURE **12** *Propagation of the stopping signal along the mauve line.*



On Figure 12, we can see that the signaling faces are already there and that they do not change. Figure 13 illustrates the stopping process. The lines are stopped one by one in this order: mauve line first, then yellow line, red line below the mauve one and at last, red line under the yellow one. In this stopping, we can note a few blank cells with signaling faces: they correspond to the previously last cells of yellow and mauve lines and a non completed cell of the red one.

In particular, in a cell of the yellow line, if face 2 or 5 is in `T`, this `T` cannot be confused with the signaling faces: indeed, assume that face 5 is in `T`. The cell might then decide that its top is face 10. Now, it would find that its back is face 2 which cannot be `T` as there cannot be two cells in `T` on the yellow line. A symmetrical argument holds if face 2 is in `T` with face 8 taken as the top. In the same way, on the mauve line if a face `D` is replaced by `dG`, this also does not prevent the identification of the wall: face 1 is recognized and as there is no more `T` on the yellow line, the single `T` face of the cell is face 0. So that, as the wall is already determined, this determines the top and, from the top, the back is also determined. Accordingly, a cell of a line always knows to which line it belongs and where is the wall which separates it from the other line sharing the same side with respect to $\mathcal{H}$. From this, the identification given by (S_3) raises no problem, where (S_3) is obtained from (S_3) by identifying _ with `W`.

_	0	1	b	c	D	T	G	H	
W	Y	pY	Y	Y	M	dR	Y	Y	dG

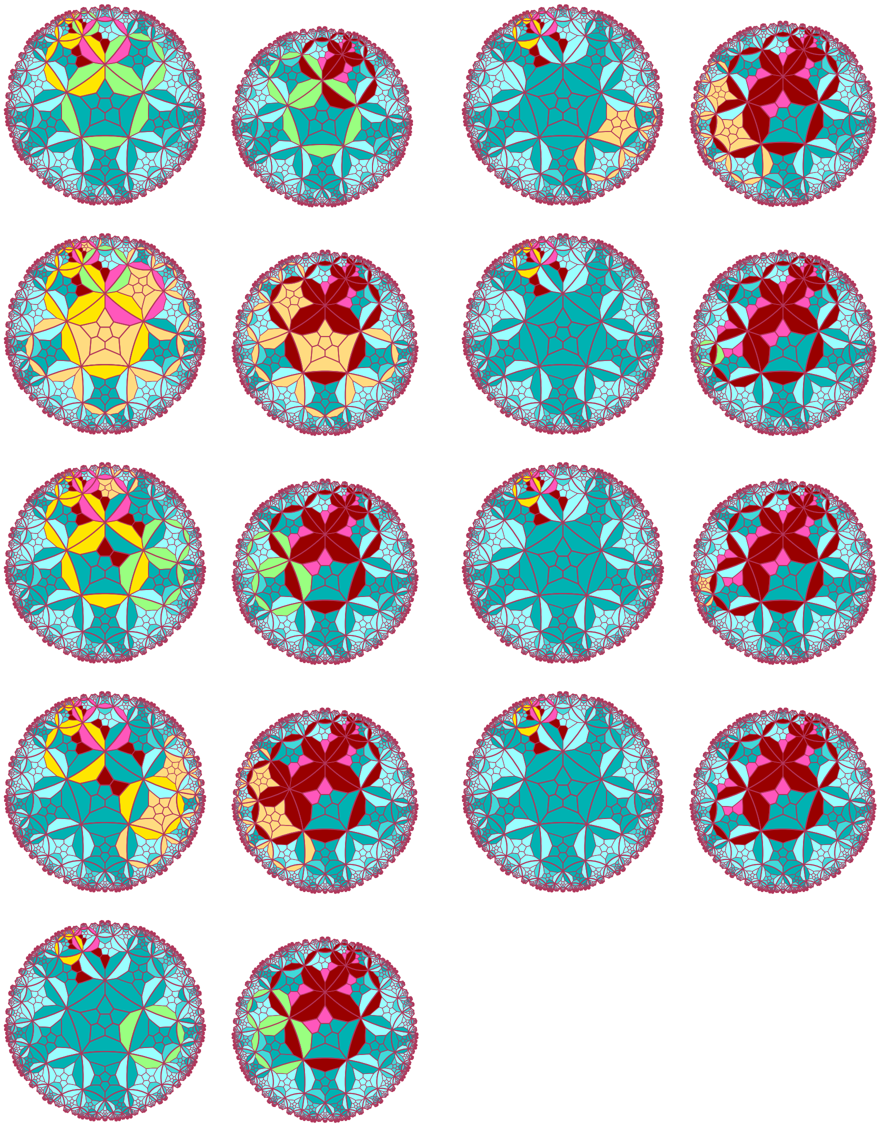
(S_3)

where _ is identified with `W`.

Accordingly we also proved Theorem 3 for the dodecagrid.

Theorem **3** *In each of the following tilings: the pentagrid and the heptagrid of the hyperbolic plane, and also in the dodecagrid of the hyperbolic $3D$-space, there is a deterministic, rotation invariant cellular automaton with radius 1 which has 10 states and which is universal.*

Figure **13** *When the stopping signal reaches the end of the mauve line.*



6 Conclusion

Theorems 2 and Theorem 3 significantly improve the result of [5, 6]. As far as known to the author, Theorem 1 is the first result on a small strongly universal cellular automaton on the line.

References

[1] M. Cook. Universality in elementary cellular automata. *Complex Systems*, 15(1):1–40, 2004.

[2] M.G. Nordahl K. Lindgren. Universal computations in simple one-dimensional cellular automata. *Complex Systems*, 4:299–318, Month 1990.

[3] M. Margenstern. *Cellular Automata in Hyperbolic Spaces, volume I : Theory*, volume 1 of *Advances in Unconventional Computing and Cellular Automata, Editor: Andrew Adamatzky.* Old City Publishing-Édition des archives contemporaines, Philadelphia, PA, USA - Paris, France, first edition, 2007.

[4] M. Margenstern. *Cellular Automata in Hyperbolic Spaces, volume II : Implementation and Computations*, volume 2 of *Advances in Unconventional Computing and Cellular Automata, Editor: Andrew Adamatzky.* Old City Publishing-Édition des archives contemporaines, Philadelphia, PA, USA - Paris, France, first edition, 2008.

[5] M. Margenstern. Universality and the halting problem for cellular automata in hyperbolic spaces: The side of the halting problem. In N. JONOSKA J. DURAND-LOSE, editor, *Unconventional Computation and Natural Computation*, volume 7445, pages 12–33. UCNC 2012, Springer, September 2012.

[6] M. Margenstern. *Small Universal Cellular Automata, A Collection of Jewels.* Emergence, Complexity and Computation. Springer, first edition, 2013.

[7] M. L. Minsky. *Computation: Finite and Infinite Machines*, volume 2 of *Advances in Unconventional Computing and Cellular Automata, Editor: Andrew Adamatzky.* Prentice Hall, Englewood Cliffs, N.J., USA, first edition, 1967.

[8] Yu. V. Rogozhin. Sem' universal'nykh mashin tjuringa. *Matematicheskie Issledovanija*, 69(2):76–90, 1982. Seven universal Turing machines (in Russian).

[9] Yu. V. Rogozhin. Small universal turing machines. *Theoretical Computer Science*, 168(2):215–240, 1996.